6.5420 Randomness and Computation	March 10, 2026
Lecture 9	
Lecturer: Ronitt Rubinfeld	*Scribe: Allen Lin*

1 Introduction

Last time, we defined pairwise independence. Recall that if we define

$$h_{a,b}(x) = ax + b \bmod q$$

for some prime q and $a, b \in \{0, \ldots, q-1\}$, then the family of functions

$$\mathcal{H} = \{h_{a,b} \mid a, b \in \mathbb{Z}_q\}$$

is pairwise independent for a random choice of $a, b \in \mathbb{Z}_q$. In other words, for a random choice of $a, b \in \mathbb{Z}_q$, $xa + b \bmod q$ and $ya + b \bmod q$ are uniformly distributed in $\mathbb{Z}_q \times \mathbb{Z}_q$ for all x and all y.

2 Error Amplification

Now we examine how pairwise independence can be used to reduce error. Suppose that we have an **RP** algorithm $\mathcal{A}$ that takes an input x and a random string r such that

- if $x \in L$, $\Pr_r[\mathcal{A}(x; r) = \text{“accept”}] > 1/2$; and
- if $x \notin L$, $\Pr_r[\mathcal{A}(x; r) = \text{“accept”}] = 0$.

There are multiple ways to reduce the confidence error of $\mathcal{A}$ to $< 2^{-k}$.

- We could run $\mathcal{A}$ a total of k times on independent bits and output 1 if and only if at least one $\mathcal{A}$ accepts. Thus uses kr random bits.
- Alternatively, we could use random walks to choose our bits. This uses $r + O(k)$ random bits.
- Today, we introdue a third method using pairwise independence that uses $O(r + k)$ random bits.

2.1 2-point Sampling

One idea is to use pairwise independent choices of random strings. In particular, we assume that given a family of pairwise independent functions $\mathcal{H}$, where each $h \in \mathcal{H}$ is a function $h\colon \left[2^{k+2}\right] \to \{0,1\}^r$, we can pick a random $h \in \mathcal{H}$ with $O(k + r)$ random bits in $\mathsf{poly}(k, r)$ time.

Then our sampling algorithm $\mathcal{B}$ is as follows.

1. Pick $h \in_R \mathcal{H}$. (This is the only place where randomness is used.)
2. For $i = 1, \ldots, 2^{k+2}$:
   (a) Set $r_i \leftarrow h(i)$.
   (b) If $\mathcal{A}(x, r_i) =$ “accept”, output “accept” and halt.
3. Output “reject.”

From our assumption on $\mathcal{H}$, we only used $O(k+r)$ random bits! However, the disadvantage of the algorithm is that its runtime is $O\left(2^k T(\mathcal{A})\right)$, where $T(\mathcal{A})$ is the runtime of $\mathcal{A}$.

If $x \notin L$, by construction the sampling algorithm never accepts, so $\Pr[\mathcal{B}(x) = \text{“accept”}] = 0$. On the other hand, if $x \in L$, then $\mathcal{B}$ rejects x only if all 2^{k+2} instances of $\mathcal{A}$ reject. Let

$$G(r_i) = \begin{cases} 0 & \text{if } \mathcal{A}(x; r_i) = \text{“reject,”} \\ 1 & \text{otherwise.} \end{cases}$$

Then

$$\mathbb{E}[G(r_i)] = \Pr[G(r_i) = 1] > 1/2.$$

Let $q = 2^{k+2}$ and

$$Y = \sum_{i=1}^{q} G(r_i).$$

Then by linearity of expectation the expected ratio of “accepts” is

$$\mathbb{E}[Y/q] \geq \frac{2^{k+2} \cdot 1/2}{2^{k+2}} = 1/2.$$

Therefore, if $x \in L$, one would expect to see at least $\geq 1/2$ “accept”s. Note that $Y \geq 1$ if and only if we find a witness, so the probability that $Y = 0$ is our desired probability. To bound this, we use the pairwise independent tail inequality.

Lemma 1 (Pairwise independent tail)**.** *Let $X_1, \ldots, X_t$ be pairwise independent random variables in $[0,1]$ with*

$$X = \frac{1}{t} \sum_{i=1}^{t} X_i$$

and $\mu = \mathbb{E}[X]$. Then

$$\Pr[|X - \mu| \geq \varepsilon] \leq \frac{1}{t\varepsilon^2}.$$

In the context of our application, we see that $\mu = \mathbb{E}[Y/q] \geq 1/2$ and $t = q$. Choosing $\varepsilon = 1/2$ then gives

$$\begin{aligned} \Pr[Y = 0] &= \Pr[Y/q = 0] \\ &\leq \Pr\left[\left|\frac{Y}{q} - \underbrace{\mathbb{E}\left[\frac{Y}{q}\right]}_{\mu}\right| \geq \underbrace{\mathbb{E}\left[\frac{Y}{q}\right]}_{\varepsilon}\right] \\ &\leq \frac{1}{t\varepsilon^2} \\ &= 2^{-(k+2)} \cdot 4 \\ &= 2^{-k}, \end{aligned}$$

which gives the required error amplification. The number of random bits used is $O(k + |R|)$.

3 Interactive Proofs

Another application of pairwise independence is found in interactive proof systems.

Recall that **NP** is the class of all decision problems for which “Yes” answers can be verified in polynomial time by a deterministic Turing machine, commonly referred to as a “verifier.” The class **IP** is a generalization of **NP**, where the existence of a short proof implies the existence of a short “interactive proof.”

3.1 The Pepsi Challenge

As an example of an interactive proof, suppose that a contestant on a game show is tasked with the challenge of identifying the difference between Pepsi and Coke. The game show develops the following procedure.

1. The game show tosses a coin (but doesn't show the contestant the result).
   (a) If the coin is heads, give the contestant Pepsi.
   (b) If the coin is tails, give the contestant Coke.
2. The contestant tastes and tells the game show "Pepsi" or "Coke."

The game show follows the above procedure k times. Now if the contestant is correct k times, then the game show is confident that the contestant can identify the difference.

- If the contestant can identify the difference, they will always get it right.
- If the contestant cannot identify the difference, the expected number of right guesses is $k/2$, and the probability that the contestant is correct k times is 2^{-k}.

Therefore, if the contestant is correct k times, either the contestant can identify the difference or is extremely lucky!

3.2 The IP model

The **IP** model consists of

- a polynomial time verifier V with random bits (may or may not be private);
- an omnipotent prover P with unbounded time but recursive (so it cannot solve the halting problem); and
- read and writing conversation tapes.

Definition 2 (Goldwasser-Micali-Rackoff)**.** *An* interactive proof system *(IPS) for a language L is a protocol such that*

- *if $x \in L$ and both V and P follow the protocol, then*

$$\Pr_{V\text{'s coins}}[V(x) = \text{“accept”}] \geq 2/3\text{; and}$$

- *if $x \notin L$ and V follows the protocol, then regardless of what P does,*

$$\Pr_{V\text{'s coins}}[V(x) = \text{“reject”}] \geq 2/3.$$

In other words, if $x \in L$, then the prover P can "convince" V. Otherwise, if $x \notin L$, then even if P "cheats" and does not follow the protocol, it cannot "convince" V.

This protocol has numerous applications.

Example 3 (Cryptography)**.** *Assume that L is a hard language to compute and $x \in L$ if and only if P is "the bank."*

- *P can convince V to trust it if it is really the bank.*
- *No impostor can convince P to trust it.*

Definition 4. *We define* **IP** *as the class of all languages L such that L has an IPS.*

Example 5 (Complexity)**.** *Clearly* $\mathbf{NP} \subseteq \mathbf{IP}$*. Let* $L \in \mathbf{NP}$*. To show* $x \in L$*, P just constructs the* **NP***-proof and sends it to V, and V verifies it. If* $x \notin L$*, there is no proof that would convince V.*

Importantly, we have the following complexity result.

Theorem 6. $\mathbf{IP} = \mathbf{PSPACE}$.

3.3 Graph Isomorphism

The graph isomorphism problem $\mathsf{GraphIsomorphism}$ is the computational problem of determining whether two graphs $G = (V_G, E_G)$ and $H = (V_H, E_H)$ are isomorphic, i.e., whether there exists an isomorphism $\pi \colon V_G \to V_H$ such that $(u, v) \in E_G$ if and only if $(\pi(u), \pi(v)) \in E_H$.

Currently, the question of determining $\mathsf{GraphIsomorphism} \in \mathbf{P}$ is open. A recent result gives a quasi-polynomial $O(n^{\mathsf{polylog}n})$ time algorithm for $\mathsf{GraphIsomorphism}$, so it is unlikely to by **NP**-complete. Moreover, $\mathsf{GraphIsomorphism} \in \mathbf{NP}$, but the question of $\overline{\mathsf{GraphIsomorphism}} \in \mathbf{NP}$ is open. However, $\overline{\mathsf{GraphIsomorphism}} \in \mathbf{IP}$, given by the following protocol to show that $G_1 \ncong G_2$.

1. V picks $C \in \{1, 2\}$ randomly.
2. V picks random relabeling of nodes in G_C and sends new adjacency matrix to P.
3. P guesses C.

Repeat this protocol k times. If $G_1 \ncong G_2$, P can guess correctly every time using its unbounded computation power. On the other hand, if $G_1 \cong G_2$, P can only guess which coin is flipped; P guesses all k coins correctly with probability 2^{-k}.

3.4 Goldwasser-Sipser Theorem

A natural question arises regarding the necessity of V's coins to be private. In the above protocol, if P saw V's coins, it could cheat. It turns out that any protocol with private coins has another protocol with public coins.

Theorem 7 (Goldwasser-Sipser)**.** $\mathbf{IP}_{private} = \mathbf{IP}_{public}$.

Today, we prove a smaller claim that is used for the theorem. Given a set $S_\varphi = \{x \mid x \text{ satisfies } \varphi\}$, we claim that there exists a protocol such that on input φ,

- if $|S_\varphi| > k$ and if V and P follow protocol, then $\Pr[V \text{ accepts}] \geq 2/3$; and
- if $|S_\varphi| < k/\Delta$ and if V follows protocol, then $\Pr[V \text{ accepts}] \leq 1/3$.

As a first idea, we consider random sampling: V picks a random assignment and evaluates $\varphi(x)$. This is repeated a set number of times and we output #satisfying x/#number of times. However, the number of repetitions is

$$\Omega\left(\frac{\#\text{total assignments}}{\#\text{satisfying assignments}}\right)$$

which could be $\Omega(2^n)$.

We now fix this with universal hashing. Recall that a family of functions $\mathcal{H}$ of the form $h_i \colon [N] \to [M]$ is pairwise independent if whenever $h \in_u \mathcal{H}$,

- for all $x \in [N]$, $h(x) \in_u [M]$; and
- for all $x_1 \neq x_2$ in $[N]$, $(h(x_1), h(x_2)) \in_u [M]^2$.

How does a pairwise independent family of hash functions help? The idea is that while $|h(S_\varphi)| \leq |S_\varphi|$, we also have $h(S_\varphi)$ is not too much smaller than S_φ. In fact, pairwise independent hash functions are typically used to

- map S into a smaller space and
- improve storage constraints by reducing the size.

Importantly, they have the property that there are few collisions, so $|h(S)|$ is not that much smaller than $|S|$. Using this idea, we can determine if a set is big with the following intuition: if y is any point and $h(S)$ is big, then $y \in h(S)$ probably. With this intuition, consider the following protocol given such a family $\mathcal{H}$.

1. V picks $h \in_R \mathcal{H}$.
2. V shares h with P.
3. P sends a point $x \in S_\varphi$ such that $h(x) = 0^\ell$.
4. V accepts if and only if $x \in S_\varphi$.

There are two regimes: $|S_\varphi| > k$ and $|S_\varphi| < k/\Delta$.

- When the size of S_φ is large (i.e., $|S_\varphi| > k$), then 0^ℓ is hit with reasonable probability. In this case, the all-powerful P can find the preimage in S_φ.
- When the size of S_φ is small (i.e., $|S_\varphi| < k/\Delta$, then 0^ℓ is very unlikely to be hit, so P mostly like cannot find the preimage. If P sends a fake preimage, then V can easily check.

We have the following important lemma.

Lemma 8. *Suppose that $\mathcal{H}$ is a pairwise independent hash family. Let $U \subseteq \{0,1\}^n$ with $a = |U|/2^\ell$. Then*

$$a - \frac{a^2}{2} \leq \Pr_{h \in \mathcal{H}}[0^\ell \in h(U)] \leq a.$$

Proof. For the upper bound, $\Pr_{h\in\mathcal{H}}[h(x) = 0^\ell] = 2^{-\ell}$ for all x since H is pairwise independent; a union bound gives

$$\Pr_{h \in \mathcal{H}}[0^\ell \in h(U)] \leq \sum_{x \in U} \Pr[h(x) = 0^\ell] = \frac{|U|}{2^\ell} = a.$$

For the lower bound, we use the inclusion-exclusion principle

$$\begin{aligned}
\Pr_{h \in \mathcal{H}}[0^\ell \in h(U)] &\geq \sum_{x \in U} \Pr[h(x) = 0^\ell] - \sum_{\substack{x,y \in U \\ x \neq y}} \Pr_{h \in \mathcal{H}}[h(x) = h(y) = 0^\ell] \\
&= \frac{|U|}{2^\ell} - \binom{|U|}{2} \frac{1}{2^{2\ell}} \\
&\geq \frac{|U|}{2^\ell} - \frac{|U|^2}{2} \cdot \frac{1}{2^{2\ell}} \\
&\geq a - \frac{a^2}{2},
\end{aligned}$$

as required. □

We now apply this lemma to our claim. There exists an ℓ such that $2^{\ell-1} \leq k \leq 2^\ell$. Let $a = |S_\varphi|/2^\ell$. We turn to our two regimes.

- If $|S_\varphi| > k$, then $a \geq 1/2$. The lower bound gives

$$\Pr[0^\ell \in h(S_\varphi)] \geq a - \frac{a^2}{2} \geq \frac{3}{8}.$$

- If $|S_\varphi| < k/\Delta$, then $a < (k/\Delta)/2^\ell < 1/\Delta$. The upper bound gives

$$\Pr[0^\ell \in h(S_\varphi)] \leq a < \frac{1}{\Delta}.$$

  If we pick $\Delta = 4$, then the upper bound becomes $1/4$.

With this analysis, it follows that if we repeat the protocol $O(\log 1/\beta)$ times, the Chernoff bound implies that with probability $\geq 1 - \beta$,

- if $|S_\varphi| > k$, then P is successful with probability $\geq 3/8 - o(1)$; and
- if $|S_\varphi| < k/4$, then P is successful with probability $\leq 1/4 + o(1)$.